CLAIMS

1. (Currently Amended) A method of calculating a checksum for a data block by reduction, the method comprising the steps of:

(a) partitioning the data block into *N* segments of a data matrix, *N* being an integer greater than one;

(b) comparing *N* to a number of segments processed by each of at least two reduction stages, the at least two reduction stages arranged in a tree structure;

(c) If *N* is less than or equal to the number of segments processed by a highest level reduction stage, then:

(1) processing the data matrix with ~~the~~ a lowest level reduction stage that is configured to ~~can~~ process the entire data matrix to generate a new data matrix, and

(2) repeating step (c)1 for each subsequent new data matrix until two data segments remain;

otherwise, if *N* is greater than the number of segments processed by the highest-level reduction stage, then:

(3) dividing the data matrix into one or more portions;

(4) processing one matrix portion with the highest-level reduction stage that is configured to ~~can~~ process the matrix portion to generate a new data matrix,

(5) repeating steps (c)(1) and (c)(2) for each subsequent new data matrix of the one matrix portion until two data segments corresponding to the one matrix portion remain,

(6) appending another ~~an other~~ portion of the data matrix to the two data segments corresponding to the one matrix portion, and

(7) repeating step (c) until no matrix portions remain; and

(d) combining the remaining two data segments to provide a result.

2. (Original) The invention as recited in claim 1, further comprising the step of inverting the result to provide the checksum of the data block.

3. (Original) The invention as recited in claim 1, further comprising the step of incrementing the result if the combination of the remaining two data segments overflows.

4. (Original) The invention as recited in claim 1, wherein:

step (c)(3) comprises the step of:

(i) dividing the data matrix into one or more portions such that the number of segments of each portion correspond to a number of segments processed by one or more of the reduction stages; and

step (c)(4) processes each matrix portion concurrently, and further comprises the step of:

(i) appending one or more new data matrices together to form a subsequent data matrix.

5. (Currently Amended) The invention as recited in claim 1, wherein step (c)(4) comprises the steps of:

i) processing one matrix portion with the highest level reduction stage that is configured to ~~can~~ process the matrix portion to generate the new data matrix;

ii) repeating step (c)(4)(i) until two matrix segments remain;

iii) appending one or more segments of an other matrix portion to the two remaining matrix segments;

iv) repeating steps (c)(4)(i)-(c4)(iv) until the two data segments remain.

6. (Original) The invention as recited in claim 1, wherein, for step (a), each segment is an *L*-bit data word.

7. (Original) The invention as recited in claim 1, wherein the data block is either a subpacket or a packet.

8. (Original) The invention as recited in claim 1, wherein the method is embodied as processing steps in a processor of an integrated circuit.

9. (Currently Amended) Apparatus for calculating a checksum for a data block by reduction, the apparatus comprising:

a processor adapted to coordinate processing of one or more reduction stages;

at least two reduction stages arranged in a tree structure, each reduction stage ~~adapted~~ configured to process a matrix in accordance with the reduction; and

a combiner adapted to combine two remaining data segments to provide a result, and

wherein:

the processor is ~~adapted~~ configured to compare i) *N* segments of a data matrix representing the

data block to ii) a number of segments processed by each of the at least two reduction stages, N being an integer greater than one, and wherein the processor is ~~adapted~~ configured to coordinate a test of:

If N is less than or equal to the number of segments processed by a highest level reduction stage, then:

(1) ~~the~~ a lowest level reduction stage that is configured to ~~can~~ process the entire data matrix processes the data matrix to generate a new data matrix, and

(2) each subsequent new data matrix is processed by one or more corresponding reduction stages until the two data segments remain;

otherwise, if N is greater than the number of segments processed by the highest-level reduction stage, then:

(3) the processor divides the data matrix into one or more portions;

(4) the highest-level reduction stage that is configured to ~~can~~ process one matrix portion processes the one matrix portion to generate a new data matrix,

(5) the processor enables repetition of (3) and (4) for each subsequent new data matrix of the one matrix portion until two data segments corresponding to the one matrix portion remain,

(6) the processor appends another ~~an other~~ portion of the data matrix to the two data segments corresponding to the one matrix portion, and

(7) repeating the test until no matrix portions remain.

10. (Currently Amended) The invention as recited in claim 9, further comprising an inverter ~~adapted~~ configured to invert the result to provide the checksum of the data block.

11. (Currently Amended) The invention as recited in claim 9, further comprising logic ~~adapted~~ configured to increment the result if the combination, by the combiner, of the remaining two data segments overflows.

12. (Original) The invention as recited in claim 9, wherein each segment is an L-bit data word.

13. (Original) The invention as recited in claim 9, wherein the data block is either a subpacket or a packet.

14. (Original) The invention as recited in claim 9, wherein the apparatus is embodied in a circuit.

15. (Original) The invention as recited in claim 9, wherein the circuit is embodied in an integrated circuit.

16. (Currently Amended) A computer-readable medium having stored thereon a plurality of instructions, the plurality of instructions including instructions which, when executed by a processor, cause the processor to implement a method for calculating a checksum for a data block by reduction, the method comprising the steps of:

(a) partitioning the data block into *N* segments of a data matrix, *N* being an integer greater than one;

(b) comparing *N* to a number of segments processed by each of at least two reduction stages, the at least two reduction stages arranged in a tree structure;

(c) If *N* is less than or equal to the number of segments processed by a highest level reduction stage, then:

(1) processing the data matrix with ~~the~~ a lowest level reduction stage that is configured to ~~can~~ process the entire data matrix to generate a new data matrix, and

(2) repeating step (c)1 for each subsequent new data matrix until two data segments remain;

otherwise, if *N* is greater than the number of segments processed by the highest-level reduction stage, then:

(3) dividing the data matrix into one or more portions;

(4) processing one matrix portion with the highest-level reduction stage that is configured to ~~can~~ process the matrix portion to generate a new data matrix,

(5) repeating steps (c)(1) and (c)(2) for each subsequent new data matrix of the one matrix portion until two data segments corresponding to the one matrix portion remain,

(6) appending another ~~an other~~ portion of the data matrix to the two data segments corresponding to the one matrix portion, and

(7) repeating step (c) until no matrix portions remain; and

(d) combining the remaining two data segments to provide a result.